Assignment 1
for Formal Languages and Automata Theory
TDDD14 and TDDD85

(2016)

For all the problems below it is not sufficient just to give a solution. **Justify your answers**. In the final exam unexplained answers will be granted 0 points. (For example, assume that you are writing a grammar for a given language. Then you should also explain that the grammar indeed generates the language). It is allowed to discuss the exercises with others, but you are supposed to solve each exercise individually. It is absolutely not allowed to copy solutions from others.

The solutions should be handed in to Jonas Wallgren or Johannes Schmidt – on paper using the postbox in front of "Java Cafe" in building B (or at a lecture or a tutorial), or in a file in an email attachment (please only plain text, or pdf).

Your solutions can be written in Swedish or English.

1. (Homework 1.4, p. 301 in the textbook). For $k \geq 1$ and $p \geq 2$, let $\Sigma = \{ 0, 1, \ldots, p-1 \}$ and

   $$A_{k,p} = \{ x \in \Sigma^* \mid x \text{ is a } p\text{-ary representation of a multiple of } k \}.$$

   Choose a pair k, p (different from 3, 10 and 3, 2) and give some strings from $A_{k,p}$ and some not belonging to $A_{k,p}$.

   Construct a DFA $M_{k,p}$ for the language $A_{k,p}$.

   A DFA $M_{3,10}$ for $A_{3,10}$ was presented at a lecture. Lecture 4 in the book gives such a DFA $M_{3,2}$ for $A_{3,2}$.

2. Consider a language

$$L = \{\, xayaz \in \{a,b\}^* \mid \ |y| = 5k, \text{ for some } k \geq 0 \,\}$$

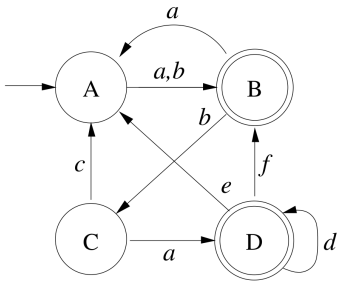
(i.e. the language of the strings over $\{a,b\}$ which have a pair of a's that are separated by a sting of length $5k$). So for instance, $aa \in L$, $b\underline{a}babbb\underline{a}bb \in L$ (look at the underlined symbols), and $bababab \notin L$.

Construct for yourself some more examples of strings in L and not in L. Which of the strings $(bbba)^i$ $(i = 0, 1, 2, \ldots)$ are in L?

Provide an NFA (or NFAε) and a regular expression for L.

Outline a way of constructing a DFA for L. (You do not need to actually construct it.) Are you able to estimate the size of the DFA? Are you able to find a number m, so that each DFA for L must have at least m states? Are you able to prove this fact, using the Myhill-Nerode theorem?

3. Consider the finite automaton given by the diagram. Is the automaton deterministic? Using a standard method construct a regular expression for the language defined by the automaton.



4. Construct a DFA equivalent to the following NFA-ϵ:

	a	b	ϵ
$\rightarrow 1$	$\emptyset$	$\emptyset$	{2,5}
2 F	{3}	$\emptyset$	$\emptyset$
3	$\emptyset$	{4}	$\emptyset$
4	$\emptyset$	{2}	$\emptyset$
5	{5}	$\emptyset$	{6}
6 F	$\emptyset$	{7}	$\emptyset$
7	{6}	$\emptyset$	$\emptyset$

5. Minimize the DFA with the set of states $\{1,2,3,4,5,6,7,8\}$, input alphabet $\Sigma = \{a,b\}$ initial state 1, final states $\{3,4\}$ and the transition function given by the table:

	a	b
$\rightarrow 1$	6	2
2	3	6
3 **F**	2	4
4 **F**	5	3

	a	b
5	4	1
6	1	5
7	1	8
8	8	7

Let L be the language accepted by this DFA. Consider the equivalence relation R_L on Σ^* (thus $R_L \subseteq \Sigma^* \times \Sigma^*$) defined by: xR_Ly if and only if for all $z \in \Sigma^*$, $xz \in L$ iff $yz \in L$. What is the index (the number of equivalence classes) of R_L? Describe precisely at least one of the equivalence classes of R_L (by giving an automaton, or a regular expression).

6. At the lectures we presented a construction of an NFAϵ M_{XY} for the concatenation $L(M_X)L(M_Y)$ of the languages accepted by two given NFAϵ. An additional intermediate state s was introduced and then ϵ-transitions from all the final states F_X of M_X to s and from s to all start states S_Y of M_Y were added. Another idea would be to glue together the set of states F_X and S_Y directly into one single state. Find out whether this idea is correct. In other words, is $L(M_X)L(M_Y)$ the language accepted by the obtained automaton, for arbitrary M_X, M_Y.

7. Prove that a language

$$\{\, z \in \{a,b,c\}^* \mid \#a(z) > 3\#b(z) \,\}$$

is not regular. Use pumping lemma or reasoning similar to the proof of the lemma.

Hint: choose a simple string to be pumped.

An alternative and more interesting version of this problem: Prove that the set of programs of a programming language chosen by you is not regular.

Hint: To make the proof manageable, you may use a homomorphism h which maps the programming language onto some simple language L'. Construct h in such a way that L' is not regular and that this fact is easy to prove. It may be convenient if h maps most of the symbols into ε.